

An Adaptive Localization System for Outdoor/Indoor Navigation for Autonomous Robots

E.B. Pacis, B. Sights, G. Ahuja, G. Kogut, H.R. Everett
Space and Naval Warfare Systems Center, San Diego
53560 Hull Street, San Diego, CA 92152

ABSTRACT

Many envisioned applications of mobile robotic systems require the robot to navigate in complex urban environments. This need is particularly critical if the robot is to perform as part of a synergistic team with human forces in military operations. Historically, the development of autonomous navigation for mobile robots has targeted either outdoor or indoor scenarios, but not both, which is not how humans operate. This paper describes efforts to fuse component technologies into a complete navigation system, allowing a robot to seamlessly transition between outdoor and indoor environments. Under the Joint Robotics Program's Technology Transfer project, empirical evaluations of various localization approaches were conducted to assess their maturity levels and performance metrics in different exterior/interior settings. The methodologies compared include Markov localization, global positioning system, Kalman filtering, and fuzzy-logic. Characterization of these technologies highlighted their best features, which were then fused into an adaptive solution. A description of the final integrated system is discussed, including a presentation of the design, experimental results, and a formal demonstration to attendees of the Unmanned Systems Capabilities Conference II in San Diego in December 2005.

KEYWORDS: robotics, autonomous navigation, localization, Kalman filtering, technology transfer

1. BACKGROUND

The first significant military use of robotic systems on the battlefield occurred in FY-04, when the Space and Naval Warfare Systems Center, San Diego (SSC San Diego) assisted the OSD Joint Robotics Program (JRP) in procuring and fielding 163 robots to address the ongoing threat of improvised explosive devices (IEDs) in Iraq and Afghanistan. The number of robotic systems grew to approximately 2000 in FY-05 and is expected to reach 4000 in FY-06. To date, however, all of these robots are strictly teleoperated devices (i.e., remote-controlled) with no onboard intelligence, and thus require intense operator involvement and high-bandwidth communications links.

To address these shortcomings, the JRP Technology Transfer Project managed by SSC San Diego seeks to enhance the functionality (ability to perform more tasks) and autonomy (with less human intervention) of these teleoperated systems.¹ The objective is to expedite advancement of the technologies needed to produce an autonomous robot that can robustly perform in battlefield situations. Instead of developing new capabilities from scratch, the approach is to assess the technology readiness levels (TRLs) of component technologies (i.e., mapping, object recognition, motion-detection-on-the-move) developed under a variety of past and ongoing R&D efforts (such as the DARPA Tactical Mobile Robot program). The most mature algorithms are integrated and optimized into a cohesive behavior architecture and then ported to various platforms used by the warfighter for further evaluation in operational environments.

Contributing sources of component technologies include the Idaho National Laboratory (INL), NASA's Jet Propulsion Laboratory, Carnegie-Mellon University (CMU), Stanford Research Institute International (SRI), University of Michigan, Brigham Young University, University of California San Diego, and University of Texas Austin, as well as other SSC San Diego projects (i.e., Man Portable Robotic System² and the ROBART series³). Starting in FY-03, the

approach was to harvest existing indoor navigation technologies developed by various players and assess their different approaches to dead reckoning, obstacle detection/avoidance, mapping, localization, and path planning. The details of these focus areas will not be discussed in this paper but can be found in previous project publications.⁴ The best features of the more promising solutions have now been integrated into an optimal system, giving an operator the ability to send an autonomous platform into an unknown indoor area and accurately map the surroundings. An *augmented virtuality* representation of the environment is derived, fusing real-time sensor information with the evolving map (Figure 1).⁶ In FY-05, the focus was expanded to include autonomous outdoor navigation, as well as additional sensor payloads for mission-specific applications (such as intruder detection). All the component technologies are integrated under an Intelligence Kernel originally developed by INL. To ensure cross-platform compatibility, the architecture is independent of the robot geometry and sensor suite, facilitating easy porting to any platform the warfighter uses. Moreover, the Intelligence Kernel allows the robot to recognize what sensors are available at any given time and adjust its behavior accordingly.

The interior mapping capability is based upon a laser-based *simultaneous localization and mapping* (SLAM) algorithm developed by the Stanford Research Institute International (SRI).⁵ SLAM enables a robot to build a map of an unknown area while dynamically determining its own position in the growing map. SRI's original implementation represented the information from the robot's wheel encoders and laser sensor as a network of probabilistic constraints linking successive poses of the robot. The encoders relate one robot pose to the next via dead reckoning, and the laser scans are matched to provide further constraints between robot poses. The use of nearby landmarks to facilitate matching successive laser scans implied that the SLAM algorithm would probably also work in outdoor areas with sufficient prominent features.

This was demonstrated as part of the Robotic Autonomous Countermining Experiment, intended to fulfill the U.S. Army Engineer School's goal to demonstrate a team of small unmanned ground vehicles (UGVs) conducting semi-autonomous countermining operations. Experiment 1 took place at INL in October 2005, where the UGV was cued to search a bounded area using overhead imagery provided by a fixed-wing UAV. The UGV successfully detected, recorded in the map, and marked on the ground the location of all buried landmines (Figure 2). The lane- and mine-marking system was designed at SSC San Diego.

The SLAM algorithm was further tested for use in urban areas. In support of the Warfighter's Associate concept,⁶ an autonomous indoor/outdoor navigation system was derived from existing component technologies, including SLAM, to allow a robot to seamlessly navigate in and out of buildings in a Military Operations in Urban Terrain (MOUT) environment. Section 2 focuses on the design and implementation of our adaptive-localization method needed to autonomously navigate in both outdoor and indoor environments. Section 3 presents experimental results during the development and optimization of the integrated system, and section 4 describes the formal demonstration in a force-protection scenario. Section 5 concludes with a discussion on planned future extensions of this work.

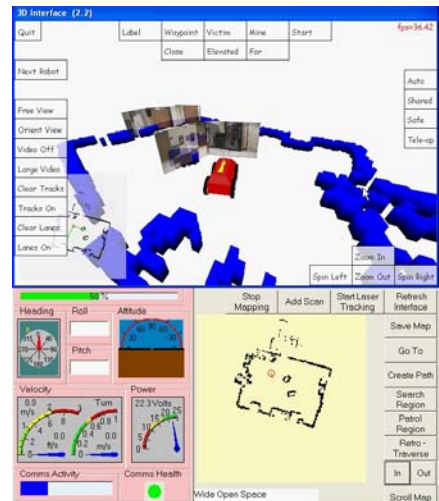


Figure 1. Virtual model of SSC San Diego lab space, fused with real-time visual images, after the robot autonomously explored and mapped the area.



Figure 2. CMU Countermine robot marked the lane and locations of the buried mines (top). The foliage along the sides of the road provided SLAM the features needed to build a map of the area searched (bottom).

2. SYSTEM DESIGN/IMPLEMENTATION OF ADAPTIVE LOCALIZATION

The extended Kalman filter (EKF) is the proverbial workhorse of the navigation sciences. We chose to use it because of the extensive knowledge already available on the subject and the ease with which it can be ported to various platforms with similar dynamic characteristics and sensor suites. Initial comparisons (discussed in section 3) between a fuzzy-logic navigation approach based on the University of Michigan's *FLEXNav* system were very promising, but the EKF was chosen when the need for cross-platform compatibility became imperative. The EKF is a well studied and extensively implemented algorithm, so we will not repeat its formulation here. For more information, please see CMU's technical report "A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles".⁷

2.1 Sensors

The EKF is a state observer that relies heavily upon the measurement of related states to produce an accurate estimate of the system state. If too few sensors are used, the resulting state estimate will diverge because the system is unobservable. Also, if the state and sensor noise variances are not accurately modeled, the EKF will not properly filter the common noise associated with each sensor, and the state estimation will further diverge from the actual state.

The key states that need to be measured for our particular implementation are: velocity, pitch, roll, and yaw rate. By measuring these four states, an accurate depiction of the robot's pose can be generated. However, using more sensors and measuring more states will create a more accurate state estimate. Therefore, the sensors used are as follows: the robot's wheel encoders, Microstrain 3DM-G gyro compass, KVH E-Core rate gyroscope, Novatel GPS, and SLAM. The states chosen to represent our system are as follows: X, Y, Z, Velocity, Pitch (θ), Roll(ϕ), Yaw(ψ), Yaw rate($\dot{\beta}$) (See Table 1).

Note that the x-axis aligns with east axis, the y-axis aligns with north, and z points upward; a yaw angle of zero degrees is associated with true north.

State	Sensor
X	GPS UTM East SLAM
Y	GPS UTM North SLAM
Velocity	Wheel encoders GPS
Pitch	3DM-G compass
Roll	3DM-G compass
Yaw	3DM-G compass GPS SLAM
Yaw speed	KVH E-Core gyroscope

Table 1. Sensors used to obtain the robot states.

2.1.1 Wheel Encoders

The wheel encoders were used to calculate the translation velocity of the robot. This type of measurement typically has errors associated with electrical noise, wheel slippage, tire size fluctuation due to temperature changes, and load shifting of the vehicle on different terrain. However, this noise was modeled simply as a fraction of the measured wheel speed (less than 5 percent) and a fraction of the measured acceleration.

2.1.2 Microstrain 3DM-G Gyro Compass

The pitch, roll, and yaw of the vehicle were measured using this sensor, which combines readings from accelerometers, gyroscopes, and magnetometers to calculate a 3-D orientation vector. The typical noise types associated with this kind of sensor are very complicated and not easily modeled. Therefore, an attitude sensor error must be garnered by referencing the manufacturer's specification sheet for the device, and it will usually contain random bias and drift. It is important to note that the yaw-axis will most likely have more error, as it is susceptible to the local magnetic field, so we doubled the yaw noise error estimate compared to the pitch and roll axes.

2.1.3 KVH E-Core Gyroscope

The KVH E-Core gyroscope is a highly accurate fiber-optic rate gyro, that can measure up to 100 degrees per second of rotational speed with a resolution of 0.014 degrees per second. This sensor needs to be accurate, as seen in Borenstein's⁸ work, which shows that the yaw estimate is crucial for accurate pose estimation. The typical error associated with a gyroscope is drift, which can be modeled as a combination of random bias, exponentially correlated error, and random walk. For robots that do not have yaw or yaw rate sensors, the wheel encoders can be used, though it is not recommended. The associated noise can be modeled as a function of the total distance traveled and the total yaw rotation.

2.1.4 Novatel GPS

The GPS unit was used to estimate the x and y coordinates of the robot by first aligning the robot local coordinates with true north. Once a reliable GPS fix has been established, a resulting offset can be calculated, which allows the GPS coordinates to be converted to local x , y coordinates based on the Universal Transverse Mercator (UTM) representation. When operating in an urban environment, the availability of GPS is typically intermittent and often suffers from the effects of occlusion and multi-path. From a dynamic standpoint, it is crucial to realize that not only will the pose estimate of the robot always have a small amount of divergence, but the GPS readings will contain drift. Since the length of time that the GPS signal will be lost is relatively short, the amount of GPS drift will be less than the divergence of the robots calculated pose. Therefore, the offset to convert the GPS coordinates to local coordinates only need to be calculated once, even if the GPS signal is lost and regained several times. The drift problem will be handled by the EKF, as are the problems associated with deciding when to input the data into the filter.

Besides providing a position estimate, the GPS unit also provides yaw and velocity measurements. However, these values cannot be trusted if the velocity of the robot is low, or if the rotational rate of the robot is greater than a couple of degrees per second. Similar to the pose estimate from the GPS unit, the decision to use the yaw and/or velocity measurements is made by the filter itself, and based on criteria described in section 2.3.

2.1.5 Simultaneous Localization and Mapping

Unlike GPS, SLAM works well indoors and in cluttered environments, but poorly in open terrain. By aligning the SLAM coordinates with world coordinates using frame transformations (further discussed in section 2.4), the output is used to measure the x , y , and yaw states of the robot. Since the errors associated with SLAM are typically random (ie. lidar errors, encoder errors, convergence errors) we simply assume the error is a constant. Currently, work is being done to automatically calculate the standard deviation of the SLAM estimate and disable and re-enable the algorithm based on this value. Also note that SLAM as an input sensor to the EKF was not used in the testing highlighted in this paper, but has since been added, properly characterized, and successfully used by the EKF algorithm. Further investigation to determine if SLAM should be an input to the EKF or if the EKF should be an input to SLAM is still ongoing.

2.2 Adaptive Filtering

A key philosophy of the Technology Transfer Program is to optimize all software to be easily portable to any robot used by the warfighter. In that light, it was necessary to allow the filter to adapt itself to the dynamics of various robots and associated sensor suites. In order to accomplish this, the Q and R matrices of the filter are alternatively re-calculated after each cycle to provide better models of the sensor and state noise. Therefore, the EKF will automatically tune itself,

and all that is needed to port the algorithm to any robot is to review a log file that is generated after the robot has been driven around for several minutes. Then the initial values of the sensor and state noise co-variances can be updated based on the converged values in the log file. The Q and R matrices are optimized as follows:

First, calculate Q_star :

$$Q_star := (X - X_k_plus_1_minus) \cdot (X - X_k_plus_1_minus)^T + P_k_plus_1_minus - P - Q$$

Then, calculate Q , where L_Q is a user-defined constant that adjusts the rate at which the adaptable filter works. This number will be different for every application:

$$Q := Q + \left(\frac{1}{L_Q} \right) \cdot (Q_star - Q)$$

As you can see, the operation to calculate Q_star takes the square of the difference between the predicted state estimate and the final state estimate of each step and subtracts the expected difference. This value is then added to a moving average of Q , which is limited by L_Q .

Now, calculate R_star :

$$R_star := (Y - H \cdot X) \cdot (Y - H \cdot X)^T - H \cdot P_k_plus_1_minus \cdot H^T$$

Then, calculate R :

$$R := R + \left(\frac{1}{L_R} \right) \cdot (R_star - R)$$

The calculation of R_star is similar to that of Q_star , but instead we take the square of the difference in the state measurements and the current state estimate and subtract the amount of expected measurement error due to the state uncertainty. This value is then added to a moving average of R , whose rate of change is limited by L_R .

This technique provides a more accurate filter with less need for time-intensive tuning by automatically adjusting the state and measurement variance matrices.⁹ It is important to note that both the Q and R matrices cannot be adapted during the same cycle. Also, using the adaptive technique while the robot is not moving for long periods of time can cause the state estimate to become unstable when the robot begins to move again. This is due to the fact that the filter is designed around a dynamic model, and it will adapt to a static model while the robot is motionless. To eliminate this problem, the adaptive technique is disabled while the robot is stationary.

2.3 Adding and Removing Sensors

Another goal in developing an enhanced dead reckoning solution was to minimize the impact on the intelligence algorithms running on the robot. This meant that we had to make the EKF smart enough to know when to add and remove sensors, and which sensors map to which states. For example, the intelligence algorithms tell the EKF that the UTM Easting and Northing measurements from the GPS sensor are available when it starts, but the EKF knows to not use these measurements until a good GPS fix has been established. This frees the higher level intelligence algorithms from having to worry about when to add and remove sensors from the EKF. Instead, the system tells the filter what kinds of sensors are available, and the EKF automatically determines which sensors are the best to use at various times. Also, the EKF must know which sensors map to which states. For example, there is not a UTM easting or northing state, so the EKF must know that these values map into the x and y states of the system. Similar processes take place with the compass yaw sensor and the GPS yaw and velocity sensors.

Another important caveat regarding adding and removing sensors is that it is bad practice to completely remove sensors from the EKF. Instead, the standard deviation for the “removed” sensor should be set arbitrarily high, which will cause the EKF to ignore the readings from the sensor. When we attempted to completely remove sensors from the EKF, we

found that the algorithm would occasionally become unstable, which was possibly due to the fact that sensors are actually coupled to several different states besides just the one they measure, so it is impossible to remove all traces of the sensor from the EKF without affecting stability.

The constraints used to determine whether or not we should use GPS as x and y state observers are as follows:

- Is the number of GPS satellites available above a certain threshold?
- Is the standard deviation being reported from the GPS unit below a certain threshold?
- What is the distance between the previous and current GPS readings (i.e., how big is the jump?), and does it make sense based on the speed travel?
- How long have all the previous criteria been met?

When GPS is used as x and y state-measurement inputs to the EKF, the following constraints are used to determine whether or not to use the yaw and velocity measurements from the GPS unit:

- Is the current velocity state of the robot above a certain threshold?
- Is the current yaw rate state of the robot below a certain threshold?

The criteria used to determine whether or not to input SLAM into to the EKF are as follows:

- What is the reported standard deviation from the SLAM algorithm?
- Are the pitch or roll states below certain thresholds? (2-D SLAM does not work well on hills)?
- Are the scans being returned from the laser showing large open spaces?
- How long have we been using GPS? (If for a long time, then the robot is probably in open terrain where SLAM is not reliable)?

2.4 Coordinate System

In order to fully comprehend our implementation and experimentation of the adaptive EKF algorithm, one must understand the coordinate space compatibility complexities behind the various sensors and integration technologies, including the communication protocol and the command-and-control system used. All raw data must be defined with the same units, otherwise their numbers are meaningless. To emphasize this point, the following are a few examples of some singular transformations that were performed to ensure the accurate representation of all data in the EKF coordinate space model.

2.4.1 EKF Model

In the EKF model, we use the local tangent plane (LTP) coordinates that represent our world as being locally flat and simplify altitude and local-direction representation for a vehicle. The specific coordinate space model used here is the east-north-up (ENU) model based on the left-handed 3-D coordinate system with the y -axis increasing going north, the x -axis increasing going east, the z -axis increasing going up, and yaw increasing counter-clockwise (Figure 3). All units are kept in the SI system with axis measurement in meters and angles in radians.

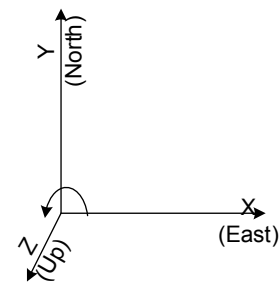


Figure 3. EKF Model uses the east-north-up (ENU) space model.

2.4.2 Microstrain 3DM-G

This orientation sensor uses a vehicle-relative coordinate space model, north-east-down (NED) (Figure 4). The 3DM-G orientation data is read as Euler Angles in degrees, providing us with pitch(θ), roll(Φ) and yaw(ψ) angles. Since we only read the orientation angles, a complete transform of the axis is not required. What is required, though, is a clear understanding of the model to ensure that the angles increase in the same direction. The only transform used is the negation of the yaw angle, which is increasing clockwise.

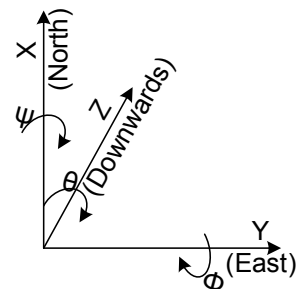


Figure 4. The compass uses a north-east-down (NED) space model.

2.4.3 Simultaneous Localization and Mapping (SLAM)

Points are defined in terms of x and y with the origin being the point where the robot first begins to navigate. The robot position is defined in terms of $\{x, y, \theta\}$, with the robot starting position as $\{0, 0, 0\}$, and x increasing in the forward direction, y increasing in the left direction, and θ increasing clockwise (Figure 5). To transform SLAM coordinates to fit our EKF model, the initial pitch values for SLAM ($\theta_SLAM_0_rad$) and EKF ($\theta_KF_0_rad$) are saved, and our transformed coordinates obtained by using the equations below:

$$rotation_angle_rad := -1 \cdot (\theta_KF_0_rad + \theta_SLAM_0_rad)$$

$$x_KF := -1 \cdot (-1 \cdot y_SLAM \cdot \cos(rotation_angle_rad) - x_SLAM \cdot \sin(rotation_angle_rad))$$

$$y_KF := -1 \cdot y_SLAM \cdot \sin(rotation_angle_rad) + x_SLAM \cdot \cos(rotation_angle_rad)$$

$$yaw_calc := \theta_KF_0_rad - (\theta_SLAM_0_rad + \theta_SLAM_rad)$$

$$yaw_calc := \begin{cases} (yaw_calc - 2\pi) & \text{if } (yaw_calc > \pi) \\ (yaw_calc + 2\pi) & \text{if } (yaw_calc < -\pi) \\ yaw_calc & \text{otherwise} \end{cases}$$

$$\theta_KF_rad := yaw_calc$$

All units in SLAM are also kept in the SI system, with axis measurements in meters and angles in radians.

2.4.4 JAUS and Multi-Operator Control Unit

The Joint Architecture for Unmanned Systems¹⁰ (JAUS) was used as the communications protocol to interface to the Multi-Robot Operator Control Unit (MOCU).¹¹ MOCU is the command and control interface used to allow the operator to select and download GPS waypoints to the robot. In order to address the low frequency bias and drift that the EKF cannot readily address, MOCU allows the operator to visually select prominent landmarks (as observed in the robot's video or aerial photograph) to be correlated with known geodetic coordinates. This generates an offset correction that is sent to the robot to compensate for the long term bias error. MOCU and JAUS both define points with the North-East-Down coordinate system, where the North position is expressed as latitude, East position as longitude, and Down represents the inverse of altitude. The coordinate transforms for this system are relatively simple, since only the z -axis is pointing in the opposite direction, but the main challenge of the transform was the unit conversions from UTM to latitude and longitude and vice-versa.

The coordinate system measurements for the EKF are in meters and our GPS data is also read in UTM. With the addition of longitude and latitude to our set of units, we had to convert our GPS data from UTM to longitude and latitude. The conversion issue does not seem difficult, with a number of different solutions available online. However, due to the nature of our experiment, where the goal was to seamlessly navigate indoors and outdoors, we had to find the most robust and consistent conversion method. The rounding of a few decimal numbers from errors after each conversion were found to displace the vehicle's position by a few meters on the map. Consequently, this might lead the operator to try to correct the already accurate position of the vehicle on the map, thereby causing the vehicle to become lost. After considerable research and trial implementations of various conversion schemes, we found that the Army Technical Manual Method¹² gave the best results. The notations used in this solution are very consistent, and the formulas are clearly laid out, making the conversions easy to implement and the output of values consistent.

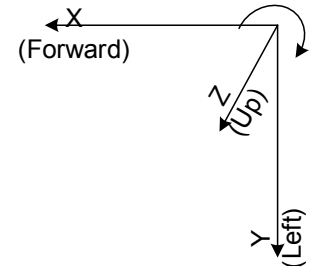


Figure 5. SLAM space model.

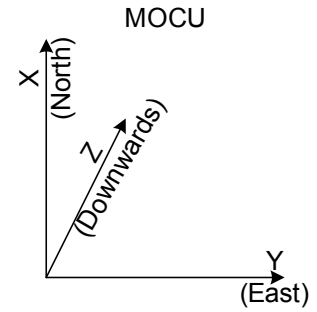


Figure 6. MOCU uses a north-east-down (NED) space model.

3. EXPERIMENTATION RESULTS

The initial tests were used to obtain a metric on the accuracy of both the fuzzy-logic implementation (based on the University of Michigan's FLEXNAV system) and the EKF implementation. Several tests were conducted over medium distances (100-300 meters) using data from the encoders for velocity, 3DM-G for pitch, roll, and yaw, and the KVH gyro for yaw rate. Markings along the path the robot was to follow were placed on the ground over the first 30 meters to allow for easy checking of the accuracy of the output of each method after leaving and returning to the same location. In order to test under a worst-case scenario, we used the iRobot *ATRV* because of its unreliable skid-steer odometry data and used non-augmented GPS (ie. non-differential GPS). The main idea was to test in a scenario that the robot may actually encounter in an urban battlefield situation. Therefore, we chose to include indoor, outdoor-urban, off-road, and open terrain settings (Figure 7) into the same test route because the robot would typically encounter all of these environments when used in theatre. These early tests showed that the fuzzy-logic implementation worked as well as, if not better than, the EKF (Figure 8). However, the fuzzy-logic implementation uses a weighted measurement system to determine which sensor to use to measure a state. It does not effectively filter the data if there is only one type of sensor to measure a given state, whereas the EKF will ignore erroneous data from any sensor. Also, there is not an easy way to provide support for any robot platform or any sensor suite with the fuzzy-logic based system. Therefore, we chose to use the EKF in our integration with the INL Intelligence Kernel, though we may implement some Fuzzy Neural Net Reasoning into the final algorithm when determining which sensors to use at different times.

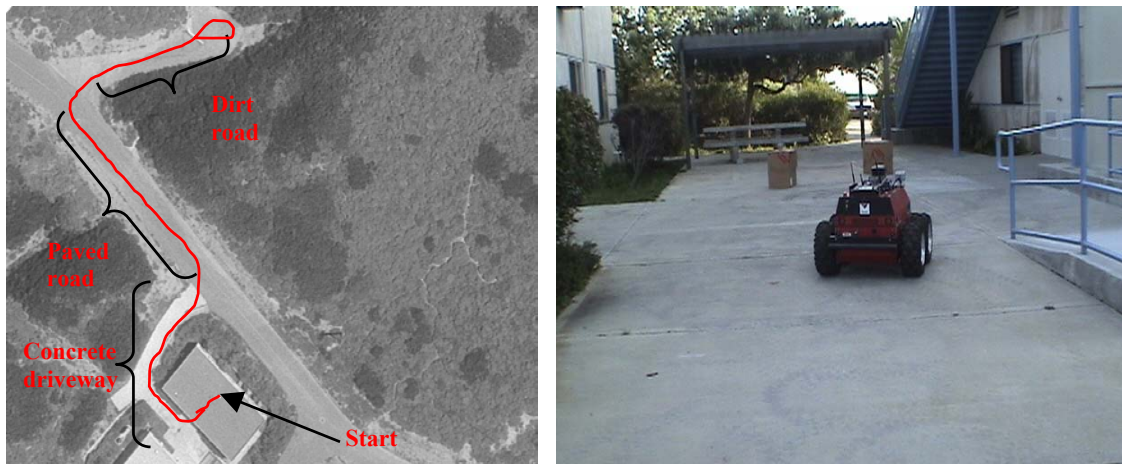


Figure 7. (Left) Aerial photograph of test site at SSC San Diego. The red line indicates the usual test route (~285m): start inside the building and exit into the courtyard between two buildings, continue up the concrete driveway, left onto the paved road, right onto the dirt road, loop around at the dead end, and return to the indoor starting position. (Right) iRobot *ATRV* navigating between the two buildings, about to return to the starting position.

After obtaining the initial metrics of each method, the data from the wheel encoders, SLAM, GPS, and the other available sensors were more precisely characterized. In order to accomplish this, tests of the robot in indoor and outdoor urban environments were run. Results (Figure 9) show that SLAM worked flawlessly, GPS provided poor data, and the EKF (with encoder, compass, and gyroscope sensor inputs) performed almost as well as SLAM.

Next, the GPS data was added into the EKF and tested in the combined indoor, urban open-space, off-road route (Figure 7) that allowed us to measure the effectiveness of each sensor across all environments. The results (Figure 10) confirmed that SLAM becomes highly unreliable in areas that do not have definitive structures that act as good landmarks (ie., buildings), but that GPS works well in these areas. It was also observed that the 2-D SLAM algorithm was adversely affected by off-road conditions, such as large pitch and roll of the vehicle and substantive slip due to the loose dirt surface. On the other hand, the EKF algorithm was stable due to the use of the yaw rate gyro and the increase in the standard deviation of the encoder velocity readings due to large accelerations when the wheels slipped.

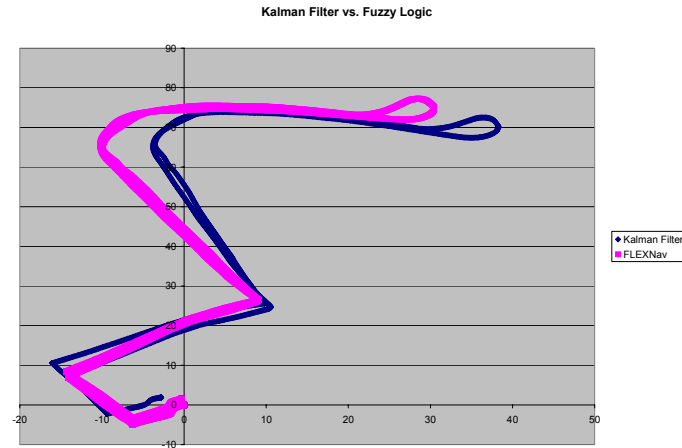


Figure 8. Results comparing an Extended Kalman Filter (blue) and a Fuzzy-logic implementation (pink) using the route in Figure 7. In this particular test run, the relative errors were 1.17 percent for the EKF and .53 percent for the Fuzzy-logic implementation. In our long term analysis, Fuzzy-logic initially produced better results, but was not as easily ported across various platforms as the EKF. Note: GPS was not used in this early experiment, and the coordinate axis was not aligned with True North.

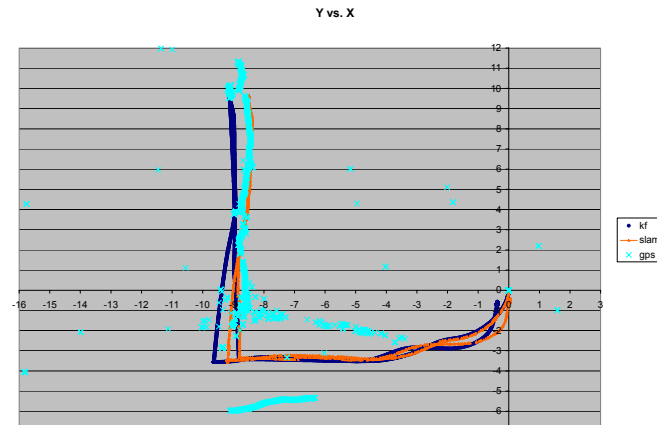


Figure 9. Test results over half the test route (Figure 7). Notice the poor GPS data (light blue), and that SLAM (red) performed slightly better than the EKF (dark blue).

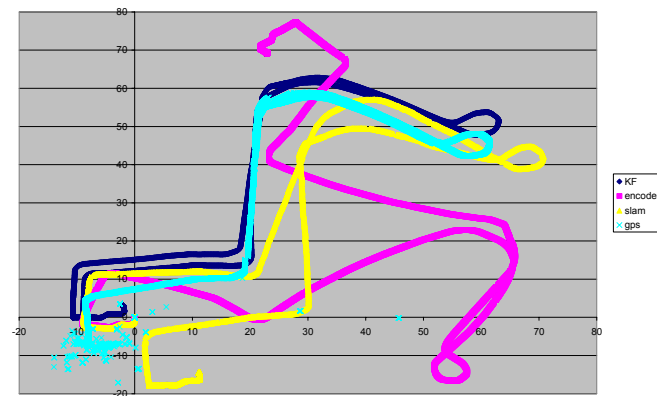


Figure 10. Test results comparing the EKF against each of the sensors alone, using the route in Figure 7. Notice how much better the EKF (blue) performed compared to SLAM alone (yellow) once the robot reached open terrain (at roughly 20m, 15m). Also notice that SLAM got off course in open terrain and on the dirt road, possibly due to the unreliable odometry data (pink) available on the ATRV. The GPS data (light blue) was very accurate while in the open and off-road terrain, but became useless in the vicinity of buildings.

Finally, the system was tested in a MOUT type scenario (Figure 11, left). This involved navigating in, near, around, and between buildings at the Unmanned Systems Branch at SSC San Diego. The EKF performed almost flawlessly (Figure 11, right), with a .35 percent relative error over a distance of 593 meters, whereas the SLAM algorithm became utterly useless after 100 meters and ended up with an error of about 45 meters. GPS was intermittently useful depending on how cluttered the environment was and how close the robot came to each building.

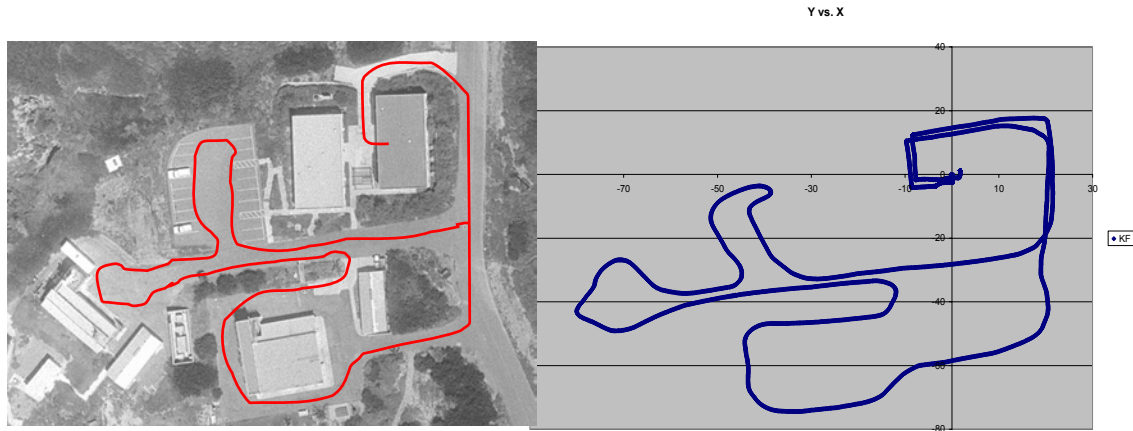


Figure 11. Overall route taken (red dotted line) to test the EKF over a 593-meter mixed urban and open-terrain course. Test results (right) showed the EKF produced a relative error of .35 percent (2.12 meters over 593 meters).

4. DEMONSTRATION

In conjunction with the Unmanned Systems Capabilities Conference II in December of 2005, SSC San Diego demonstrated a series of collaborative behaviors of multiple autonomous robots in a force-protection scenario. Stand-alone sensors detected intruder activity along the shoreline, which prompted the deployment of a USV, UAV, and UGV to further investigate. The UGV (equipped with the adaptive localization method and all the other autonomous behaviors hosted under the Intelligence Kernel) was sent into an underground bunker to search for the intruders. An optimal path of approach was first created by the operator selecting a series of waypoints on the aerial map display from MOCU, based on recently downloaded overhead imagery.

The UGV was then dispatched to the insertion point by executing autonomous waypoint navigation and collision avoidance under remote operator supervision. Upon arrival at the insertion point, the UGV seamlessly transitioned to autonomous navigation mode, relying solely on SLAM as GPS ceased to function due to satellite occlusion. A real-time communications data link was not required as the robot penetrated deeper into the bunker since the robot can continue to execute its mission in autonomous fashion. The intruder was detected (in motion and stationary), as well as a .50-calibre machine gun left behind, by utilizing a *real-time occupancy change analyzer*.¹³ When communications was restored, a copy of the virtual model of the bunker including visual snapshots of the detected intruder and weapon was passed back to the operator. Upon completion of the structure sweep, the robot planned a path back to the entrance and exited the bunker, switching back to GPS waypoint navigation. Figure 12 illustrates the sequence of events during the demonstration.

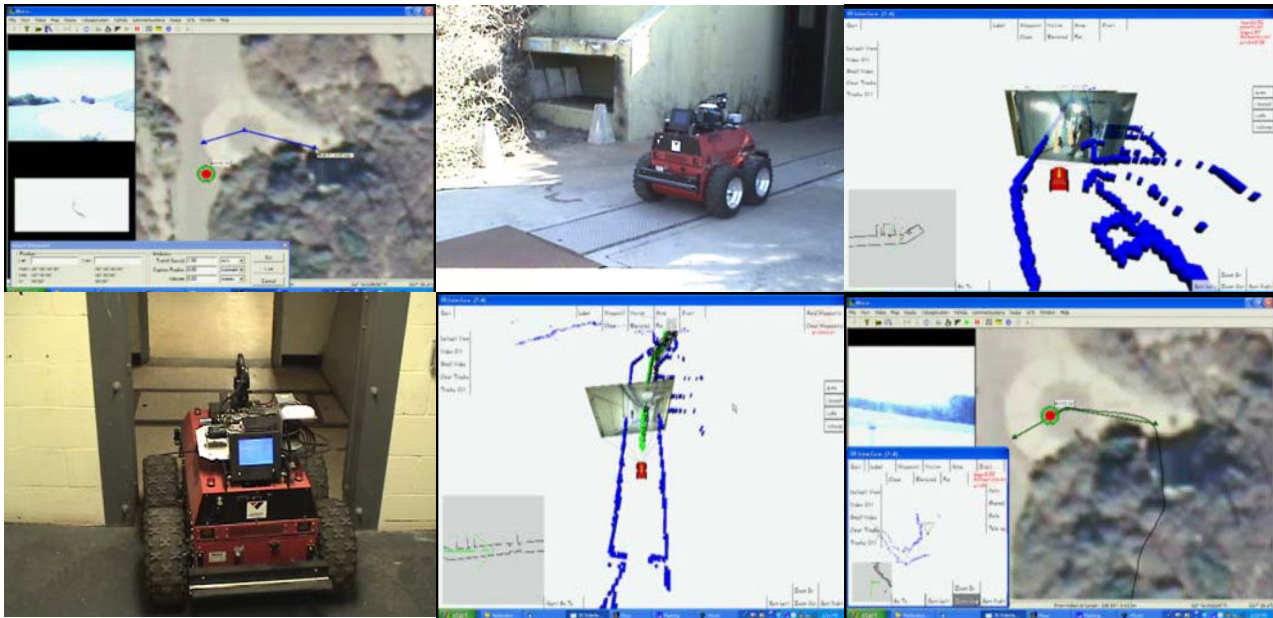


Figure 12. Demonstration of autonomous outdoor/indoor navigation and human-presence detection. Left to right: 1) GPS waypoint path sent to *ATR*V via MOCU to enter the bunker, 2) *ATR*V moving from the open outdoor environment to the bunker entrance, 3) augmented virtuality interface embedded with location tags (orange cones) of anomalies found and visual capture of intruder, 4) *ATR*V penetrating the corridors autonomously, 5) path planned (green) by robot to exit bunker autonomously. 6) aerial view showing robot's executed path outside and inside the bunker.

5. CONCLUSION AND FUTURE WORK

This paper presented a practical navigation solution that can enable a robotic platform to operate alongside a soldier in an urban setting. An adaptive localization method was developed to fuse component technologies for autonomous navigation in and out of buildings. One of the main challenges was to formulate the best algorithm for determining when to use each type of sensor depending on the type of operating environment. Extensive work has also been done to transform all the sensor and localization algorithms into a common coordinate space model. The key accomplishment here is, for the first time, the ability to geo-reference any sensor data from an autonomous platform while it is navigating indoors, which to our knowledge has not been previously done.

Advances to the SLAM algorithm have been made since the early testing described in this paper that allow input of the EKF data and output GPS position, which may facilitate a more useful and reliable navigation system. In the meantime, methods of re-initializing SLAM when the robot determines it is in an urban environment, and disabling and saving the output from the SLAM algorithm when entering open terrain, are being investigated. As humans naturally use their vision to localize themselves, machine vision for localization based on camera pose estimation is also under evaluation for future integration.

Our effort towards autonomous urban navigation is, in many respects, a "Holy Grail" of unmanned vehicle use in battlefield situations. As more and more soldiers desire the use of a robot in life-threatening situations, the requirement for proximal operations is inevitable. Solving the urban navigation problem opens up the door for further integration of autonomous capabilities and natural human-interface methods to realize soldier-robot teams.

REFERENCES

-
- ¹ Pacis, E.B, Everett, H.R., Farrington, N., Kogut, G., Sights, B., Kramer, T., Thompson, M., Bruemmer, D., and D. Few, "Transitioning Unmanned Ground Vehicle Research Technologies," SPIE Proc. 5804: Unmanned Ground Vehicle Technology VII, Orlando, FL, March 29-31, 2005.
- ² Bruch, M.H., Gilbreath, G.A., Muelhauser, J.W., and J.Q. Lum, "Accurate Waypoint Navigation Using Non-differential GPS," AUVSI Unmanned Systems 2002, Lake Buena Vista, FL, July 9-11, 2002
- ³ <http://www.spawar.navy.mil/robots/land/robart/robart.html>
- ⁴ <http://www.spawar.navy.mil/robots/pubs/PubsIdx.html>
- ⁵ D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots," Sixteenth National Conference on Artificial Intelligence (AAAI'99), July, 1999
- ⁶ Everett, H.R., Pacis, E.B, Kogut, G., Farrington, N., and S. Khurana, "Towards a Warfighter's Associate: Eliminating the Operator Control Unit," SPIE Proc. 5609: Mobile Robots XVII, Philadelphia, PA, October 26-28, 2004.
- ⁷ A. Kelly, "A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles", 1994, Carnegie Mellon University
- ⁸ Borenstein, J., L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots." IEEE Transactions on Robotics and Automation, Vol. 12, No. 6, December 1996.
- ⁹ Busse, F.D. (Stanford University), How, J.P. (MIT), J. Simpson (NASA Goddard Space Flight Center), "Demonstration of Adaptive Extended Kalman Filter for Low Earth Orbit Formation Estimation Using CDGPS," Institute of Navigation GPS Meeting, Portland, OR, September 2002.
- ¹⁰ "Joint Architecture for Unmanned Systems," Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics. <<http://www.jauswg.org/>>.
- ¹¹ Bruch, M. H., "The multi-robot operator control unit (MOCU)," SPIE Proc. 6230: Unmanned Systems Technology VIII, Defense Security Symposium, Orlando, FL, April 18-20, 2006.
- ¹² Army, Department of, 1973, Universal Transverse Mercator Grid, U.S. Army Technical Manual TM 5-241-8, p 64.
- ¹³ Kogut, G., Ahuja, G., Pacis, E.B., Carroll, D., Giles, J., Rogers, Dr. B, Nanzer, J, "Sensor Fusion for Automatic Detection of Human Presence", 2006 International Joint Topic: 9th Emergency Preparedness and Response/11th Robotics and Remote Systems for Hazardous Environments, Salt Lake City, Utah, February 12-15, 2006.